WE CLAIM:

1. A method of debugging a plurality of distributed programs, comprising:

identifying a plurality of processes;

initializing each of said processes;

executing with a single thread of control among said processes; and

continuously switching between said processes to obtain status information relating thereto.

2. The method of Claim 1, wherein the act of identifying a plurality of processes comprises identifying at least one simulation process and at least one hardware process.

3. The method of Claim 2, further comprising analyzing said status information to identify at least one or more occurrences or errors within at least one of said distributed programs.

4. The method of Claim 2, further comprising:

defining at least one object class,

defining at least one first object subclass for said at least one hardware process; and

defining at least one second object subclass for said at least one simulation process.

5. The method of Claim 1, wherein the act of executing comprises providing at least one first instance variable adapted for control of at least one of said plurality of processes.

6. The method of Claim 5, further comprising dynamically changing polling times associated with said at least one of said plurality of processes based on the status thereof.

7. The method of Claim 5, further comprising defining an interface to a first library for said at least one hardware process.

8. The method of Claim 7, further comprising accessing said first library via said interface in order to provide functions relating to at least one extension instruction.

9. The method of Claim 8, further comprising defining an interface to a first library for said at least one simulation process.

10. The method of Claim 8, further comprising accessing said first library via said interface in order to provide functions relating to at least one extension instruction, including the implemention of said at least one extension instruction.

11. The method of Claim 2, wherein the act of continuously switching comprises cycling between said processes in repeated succession.

12. The method of Claim 2, wherein the act of initializing comprises:

initializing a first process resident on a first hardware processor; and

initializing a second process on a simulator.

13. The method of Claim 1, further comprising:

defining a plurality of individual subclasses for each of said plurality of processes; and

implementing at least a portion of said subclasses as a dynamically loadable library.

14. A system for debugging heterogenous processors, comprising:

a processor having at least one debug process running thereon, and at least one simulation process associated and in data communication with said at least one debug process; and

at least one hardware process in data communication with said at least on debug process;

wherein said at least one simulation and hardware processes are controlled via a single thread.

15. The system of Claim 14, wherein said processor comprises a digital processor embodied as an integrated circuit, and said at least one debug process comprises a computer program adapted to run on said integrated circuit.

16. The system of Claim 15, further comprising at least one external port adapted for said data communication with respective ones of said at least one hardware processes.

17. The system of Claim 16, wherein said at least one simulation process comprises a computer program running on said digital processor.

18. The system of Claim 17, further comprising a plurality of dynamically loadable libraries, at least a portion of said libraries being adapted for communication with said at least one debug process.

19. A storage device adapted for use with a computing device, comprising:

a storage medium configured to store a plurality of data thereon; and

a plurality of data stored thereon, said data comprising a computer program adapted to run on a processor, and configured to debug one or more other computer programs using the method comprising:

identifying a plurality of processes;

initializing each of said processes;

executing with a single thread of control among said processes; and

continuously switching between said processes to obtain status information relating thereto.

20. A method of debugging a plurality of distributed programs, comprising:

identifying a plurality of processes;

defining at least one interface between a debug process and said plurality of processes;

initializing each of said processes;

executing with a single thread of control among said processes; and

selectively permitting at least a portion of said processes to operate while stopping others of said processes using a single one of said at least one interface.

21. A method of optimizing the operation a multi-processor system comprising a plurality of software processes, comprising:

initializing each of said processes;

executing with a single thread of control among said processes;

iteratively obtaining execution profiling information from at least a portion of said processors; and

optimizing the operation of said system based at least in part on said execution profiling information.

22. A method of coordinating the operation of at least one simulation process with at least one hardware process, comprising:

initializing at least one simulation process;

initializing at least one hardware process;

defining at least one interface between a debug process and said plurality of processes;

executing with a single thread of control among said processes; and

controlling said at least one simulation process and said at least one hardware process via said at least one interface.

23. A method of debugging a plurality of processors using a debug process, said debug process being in data communication with said processors, comprising:

initializing each of said processors;

executing with a single thread of control among said processors using said debug process;

establishing a minimum polling time for each of said processors; and

obtaining status information from each of said processors based at least in part on said polling interval.

24. An apparatus adapted for debug of a plurality of heterogeneous processors, comprising:

a digital processor;

at least one debug process adapted to run on said digital processor; and

a plurality of software processes distributed among said plurality of heterogeneous processors;

wherein said plurality of processes are adapted to gather status information regarding respective ones of said heterogeneous processors as controlled by said debug process.